Παραδείγματα με δομές, αρχεία και δυναμική διαχείριση μνήμης

Α. Πρόγραμμα για τη δυναμική δημιουργία διάταξης n δομών (stud), το διάβασμα της διάταξης και την αποθήκευση της σε δυαδικό αρχείο.

```
#include <stdio.h>
#include<stdlib.h>
struct stud {
            int id;
            char name[30];
            int mark;
};
main(){
  FILE *fp = NULL;
  struct stud *ptr;
  int i,n;
  printf("Enter n: "); //πόσες δομές?
  scanf("%d",&n);
// Δυναμική δέσμευση μνήμης για η δομές τύπου stud- διάταξη n δομών
  ptr=(struct stud*)malloc(n*sizeof(struct stud));
  for(i=0;i<n;++i){  // διάβασμα των n δομών τυπου stud - διάταξη
       printf("Student %d ===\n",i+1);
       printf("    Enter name:");
       scanf("%s",ptr[i].name); fflush(stdin); //   scanf("%s",&(ptr+i)->name);  //εναλλακτικά
       printf("    Enter ID:");
       scanf("%d",&ptr[i].id);     // scanf("%d",&(ptr+i)->id); //εναλλακτικά, με συμβολισμό pointer
       printf("    Enter Mark:");
        scanf("%d",&ptr[i].mark); // scanf("%d",&(ptr+i)->mark); //εναλλακτικά με συμβολισμό pointer
  }
  printf("\nDisplaying Info:\n");//εμφάνιση στην οθόνη των δομών τύπου stud της διάταξης
  for(i=0;i<n;++i)
  //  printf("%s\t%d\t%d\n",(ptr+i)->name,(ptr+i)->id,(ptr+i)->mark );  //με συμβολισμό pointer
       printf("%s\t%d\t%d\n",ptr[i].name,ptr[i].id,ptr[i].mark );
  // Άνοιγμα δυαδικού αρχείου για την αποθήκευση της διάταξης των δομών τύπου stud
  if ((fp=fopen("struc_stud.b", "wb+")) != NULL)    {
        for(i=0;i<n;++i)
                fwrite((ptr+i), sizeof(stud), 1, fp);
        rewind(fp);  // μετάβαση στην αρχή του αρχείου
        for(i=0;i<n;++i)      //διάβασμα της διάταξης από το δυαδικό αρχείο
                fread((ptr+i), sizeof(stud), 1 , fp);
        printf("\nWe got from file:\n");
        for(i=0;i<n;++i)
                printf("%s\t%d\t%d\n",(ptr+i)->name,(ptr+i)->id,(ptr+i)->mark );
  }
  else printf("Error opening the file struc_stud.b\n");
  free(ptr);
  return 0;
}
```

A. Πρόγραμμα για τη δυναμική δημιουργία διάταξης n δομών (stud), το διάβασμα της διάταξης με συνάρτηση, την αύξηση στο μέγεθος της διάταξης κατά m δομές, το διάβασμα των νέων δομών της διάταξης και την αποθήκευση της διάταξης σε δυαδικό αρχείο.

```
#include <stdio.h>
#include<stdlib.h>
struct stud {
            int id;
            char name[30];
            float mark;
};
struct stud read_student( );   //function to read a stud structure
void read_student_p(struct stud *); //different  function to read a stud structure
void print_student (struct stud ) ; //function to print a stud structure

main(){
  FILE *fp = NULL;
  struct stud *ptr, *tmp;
  int i,n,m;
  printf("Enter how many students? ");
  scanf("%d",&n);
// Allocates the memory for n stud structures (array) with pointer ptr pointing to the base address.
  ptr=(struct stud*)malloc(n*sizeof(struct stud));   //no error detection!
  for(i=0;i<n;++i){
       ptr[i] = read_student();
//      *(ptr+i) = read_student();  //using pointer notation
  }
  printf("\nDisplaying Info:\n");
  for(i=0;i<n;++i){
        print_student(ptr[i]);  //call function to printout the structure
        printf ("\n");
  }
  printf("How many more? "); //asks for more entries
  scanf("%d",&m);
  //reallocates m more stud structures to a  temporal pointer (tmp)
  tmp = (struct stud*) realloc(ptr, (n+m)*sizeof(struct stud));
  if (tmp ==NULL) {
                    printf ("realloc: reallocation error");
                    exit (1);
  }
  else ptr=tmp;  //since no error give ptr the value of tmp
  for(i=0;i<m;++i){      //reads the new records   - uses function read_student_p()
              read_student_p(&ptr[n+i]);
            //read_student_p(ptr+n+i);   //usring pointer notation

  }
  for(i=0;i<n+m;++i){
      print_student(ptr[i]);
      printf ("\n");
  }
  n=n+m;
```

```
// Opens binary file to write the ptr array
  if ((fp=fopen("struc_stud.b", "wb+")) != NULL){
      for(i=0;i<n;++i)
          fwrite((ptr+i), sizeof(stud), 1, fp);
      rewind(fp);  //go at the beginning of the file
      for(i=0;i<n;++i)    //read the file
         fread((ptr+i), sizeof(stud), 1 , fp);
      printf("\nReading from file:\n");
      for(i=0;i<n;++i) //prints out what written
        printf("%s\t%d\t%0.2f\n",(ptr+i)->name,(ptr+i)->id,(ptr+i)->mark ); //using pointer notation
       //  printf("%s\t%d\t%0.2f\n", ptr[i].name,ptr[i].id,ptr[i].mark );   //using index notation
  }
  else printf("Cannot open file struct_stud.b");
  free(ptr);
  return 0;
}

struct stud read_student( ){   //returns a stud structure
   struct stud student2;
   printf("Student's ID:");scanf("%d", &student2.id);
   printf("Student's name:");scanf("%s",student2.name);
   printf("Student's mark:");scanf("%f",&student2.mark);
   return (student2);
}
void print_student (struct stud student2){   //prints out  a stud structure
           printf( "ID is: %d, ", student2.id);
           printf( "name is: %s ", student2.name);
           printf( "marks are: %.2f ", student2.mark);
}
void read_student_p(struct stud *student2){    //fills a stud structure using its pointer
    printf("Student's ID:");scanf("%d", &student2->id);
    printf("Student's name:");scanf("%s",student2->name);
    printf("Student's mark:");scanf("%f",&student2->mark);
}
```